An Analysis of the (μ+1) EA on Simple Pseudo-Boolean Functions

Carsten Witt*

FB Informatik, LS 2, Univ. Dortmund, 44221 Dortmund, Germany
carsten.witt@cs.uni-dortmund.de

Abstract. Evolutionary Algorithms (EAs) are successfully applied for optimization in discrete search spaces, but theory is still weak in particular for population-based EAs. Here, a first rigorous analysis of the (μ+1) EA on pseudo-Boolean functions is presented. For three example functions well-known from the analysis of the (1+1) EA, bounds on the expected runtime and success probability are derived. For two of these functions, upper and lower bounds on the expected runtime are tight, and the (μ+1) EA is never more efficient than the (1+1) EA. Moreover, all lower bounds grow with μ. On a more complicated function, however, a small increase of μ provably decreases the expected runtime drastically.

For the lower bounds, a novel proof technique is developed. The stochastic process creating family trees of individuals is investigated and relationships with well-known models of random trees, e. g., uniform random recursive trees, are established. Thereby, a known theory on random trees is transferred to the analysis of EAs. Moreover, generalizations of the technique are applicable to more complex population-based EAs.

1 Introduction

Evolutionary Algorithms (EAs) are successfully applied to optimization tasks, but a solid theoretical foundation with respect to their computational time complexity is still missing. Runtime analysis of EAs often focuses on simple single-individual EAs such as the (1+1) EA (e. g., Garnier, Kallel and Schoenauer (1999), Droste, Jansen and Wegener (2002)). Regarding population-based EAs, runtime analyses exist for the case of multi-objective optimization (Giel (2003)), but there the purpose of a population is different than in the single-objective case. We consider discrete search spaces and single-objective optimization, in particular the maximization of pseudo-Boolean functions $f\colon \{0,1\}^n \to \mathbb{R}$. Here, runtime analyses of crossover-based EAs (e. g., Storch and Wegener (2003), Jansen and Wegener (2001c)) and of steady-state EAs using fitness-proportional selection and mutation only (Jansen and Wegener (2001b), Witt (2003)) are known. However, analyses for standard (μ+λ) EAs using uniform selection for reproduction are rare for $\mu > 1$. Up to now, there are only results on (1+λ) EAs

* supported by the Deutsche Forschungsgemeinschaft (DFG) as a part of the Collaborative Research Center "Computational Intelligence" (SFB 531)

K. Deb et al. (Eds.): GECCO 2004, LNCS 3102, pp. 761–773, 2004.

(Jansen and De Jong (2002)) and some variants of (μ+μ) EAs (He and Yao (2002)).

The aim of this paper is to contribute to a theory of standard (μ+λ) EAs, where $\mu > 1$. We start with the simple case $\lambda = 1$, considering a (μ+1) EA that is a generalization of the (1+1) EA for the search space $\{0,1\}^n$, and follow the research line started for this (1+1) EA. We study the behavior of the (μ+1) EA on example functions and compare the obtained results with those for the (1+1) EA. To this end, a new and general proof technique for bounding the expected runtime of the (μ+1) EA from below is developed. An advantage of the new method is that it has not been designed for a special mutation operator. In particular, we are able to analyze the (μ+1) EA with a global search operator that may flip many bits. Often, the analysis of EAs is much more difficult with a global than with a local search operator (see, e. g., Wegener and Witt (2003)).

The paper is structured as follows. In Sect. 2, we define the (μ+1) EA and the considered example functions. Moreover, we introduce the tool of family trees, which is essential throughout the paper. In Sect. 3, simple upper bounds on the expected runtime of the (μ+1) EA on the example functions are presented. In Sect. 4, we describe the new lower bound technique completely but omit the proofs of technical lemmas due to space limitations; a full version of the paper is available as a technical report. In Sect. 5, we apply the technique to prove lower bounds on the expected runtime and bounds on the success probability. These bounds are tight for two of the examples. Moreover, they show that here the (μ+1) EA is never more efficient than the (1+1) EA. However, it is a common belief that a population helps to better explore the search space, and it is important to find an example where the (μ+1) EA with $\mu > 1$ outperforms the (1+1) EA. Therefore, a function where an increase of μ by a sublinear factor decreases the expected runtime drastically, namely from exponential to polynomial, is identified in Sect. 6. We finish with some conclusions.

2 Definitions

We obtain the (μ+1) EA for the maximization of functions $f\colon \{0,1\}^n \to \mathbb{R}$ as a generalization of the well-known (1+1) EA (see Droste, Jansen and Wegener (2002)). As for continuous search spaces, a pure (μ+1) evolution strategy should do without recombination and should employ a uniform selection for reproduction. As usual, a truncation selection is applied for replacement. The mutation operator should be able to search globally, i. e., to flip many bits in a step. Therefore, a standard mutation flipping each bit with probability $1/n$ seems the most sensible. These arguments lead to the following definition of the (μ+1) EA.

1. Choose μ individuals $x^{(i)} \in \{0,1\}^n$, $i \in \{1,\dots,\mu\}$, uniformly at random. Let the multiset $X^{(0)} = \{x^{(1)},\dots,x^{(\mu)}\}$ be the initial population at time 0.
2. Repeat infinitely
   a) Choose an x from the population $X^{(t)}$ at time t uniformly at random.
   b) Create x' by flipping each bit of x independently with probability $1/n$. Let X' be the population obtained by adding x' to $X^{(t)}$.

c) Create $X^{(t+1)}$, the current population at time $t+1$, by deleting an individual from X' with lowest f-value uniformly at random. Set $t := t+1$.

We have kept the (μ+1) EA as simple as possible and refrain from employing diversity-maintaining mechanisms. The (μ+1) EA with $\mu = 1$ is very similar to the (1+1) EA, but differs in one respect. If an individual created by mutation has the same f-value as its father, either of both is retained with equal probability.

As usual in theoretical investigations, we leave the stopping criterion of the (μ+1) EA unspecified. We analyze the number of iterations (also called *steps*) of the infinite loop until the current population for the first time contains an optimal individual, i. e., one that maximizes f. The sum of this number and the population size μ is denoted as the *runtime* of the (μ+1) EA and corresponds to the number of function evaluations (a common approach in black-box optimization, cf. Droste, Jansen, Tinnefeld and Wegener (2002)). Throughout the paper, we consider only $\mu = \mathit{poly}(n)$, i. e., values of μ bounded by a polynomial of n.

We study the (μ+1) EA on the following example functions. The well-known function $\textsc{OneMax}(x) = x_1 + \cdots + x_n$ counts the number of ones of a string $x \in \{0,1\}^n$ and $\textsc{LeadingOnes}(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_j$ counts the number of leading ones. The function $\mathrm{SPC}(x)$ *(short path with constant fitness)* introduced by Jansen and Wegener (2001a) equals $n - \textsc{OneMax}(x)$ if x cannot be written as $1^i 0^{n-i}$ for any i. It equals $2n$ if $x = 1^n$ and $n+1$ otherwise. SPC is of special interest since EAs have to cross a plateau of constant fitness to find the optimum.

To elucidate the utility of the (μ+1) EA's population, throughout the paper, we compare the (μ+1) EA with μ parallel runs of the (1+1) EA. The total cost (neglecting initialization cost) of t steps of the (μ+1) EA corresponds to the cost raised by μ parallel runs of the (1+1) EA up to time t/μ. Thus, if we consider a (μ+1) EA at time t, we denote μ parallel runs of the (1+1) EA considered at time t/μ as the *corresponding parallel run*. In order to derive runtime bounds for the (μ+1) EA, it is helpful to consider the so-called *family trees* of the individuals from the initial population (this concept has been introduced in a different context by Rabani, Rabinovich and Sinclair (1998)). Fix an arbitrary such individual x. If x is mutated, a descendant of x is produced. More generally, we can visualize the descendants of x and their descendants by the family tree $T_t(x)$ at time t as follows. $T_0(x)$ contains only the root x. $T_t(x)$ contains $T_{t-1}(x)$ and the additional edge $\{v, w\}$ if w is the result of a mutation of the individual v at time $t-1$ and v is contained in $T_{t-1}(x)$. Note that the tree $T_t(x)$ may contain individuals that have already been deleted from the corresponding population.

3 Upper Bounds

The following upper bounds on the runtime are not too difficult to obtain.

Theorem 1. *Let $\mu = \mathit{poly}(n)$. Then the expected runtime of the (μ+1) EA on* LeadingOnes *is bounded above by $\mu + 3en \cdot \max\{\mu \ln(en), n\} = O(\mu n \log n + n^2)$.*

Proof. We measure the progress to the optimum by the potential L, defined as the maximum LeadingOnes value of the current population's individuals. To increase L, it is sufficient to select an individual with maximum value and to flip the leftmost zero. The selection and the mutation operator of the (μ+1) EA are independent. Hence, if there are i individuals with maximum value, the probability of the considered event is at least $\frac{i}{\mu} \cdot \frac{1}{n} \cdot \left(1-\frac{1}{n}\right)^{n-1} \geq \frac{i}{\mathrm{e}\mu n}$, and the waiting time is at most $\mathrm{e}\mu n/i$. The potential has to increase at most n times. Estimating $i \geq 1$ would lead to an upper bound $\mu+\mathrm{e}\mu n^2$ on the expected runtime.

However, the (μ+1) EA can produce replicas of individuals with maximum function value. If their number is i, the probability of creating a further replica is at least $(i/\mu)(1-1/n)^n \geq i/(2\mathrm{e}\mu)$. Furthermore, if $i < \mu$, this replica replaces a worse individual and increases the number of best ones. Assume pessimistically that L stays fixed until we have at least $\min\{n/\ln(\mathrm{e}n), \mu\}$ replicas. The expected time for this is, by elementary calculations, at most $2\mathrm{e}\mu \ln(\mathrm{e}n)$. Now the expected time to increase L is at most $\mathrm{e}\mu n/(\min\{n/\ln(\mathrm{e}n), \mu\})$. Altogether, the expected runtime is at most $\mu+n(2\mathrm{e}\mu \ln(\mathrm{e}n)+\frac{\mathrm{e}\mu n}{\min\{n/\ln(\mathrm{e}n),\mu\}}) \leq \mu+3\mathrm{e}n\cdot\max\{\mu \ln(\mathrm{e}n), n\}$. □

Theorem 2. *Let $\mu = poly(n)$. Then the expected runtime of the (μ+1) EA on* OneMax *is bounded above by $\mu + 5\mathrm{e}\mu n + \mathrm{e}n \ln(\mathrm{e}n) = O(\mu n + n \log n)$.*

Proof. The proof idea is similar as in Theorem 1. Let L be the maximum OneMax value of the current population. In contrast to LeadingOnes, the probability of increasing L depends on L itself. Since each individual has at least $n-L$ zeros, the considered probability is bounded below by $\frac{i}{\mu} \cdot \frac{n-L}{n} \cdot \left(1-\frac{1}{n}\right)^{n-1} \geq \frac{i(n-L)}{\mathrm{e}\mu n}$ if the population contains at least i individuals with maximum value.

The expected time until the population contains at least $\min\{n/(n-L), \mu\}$ replicas of an individual with value L is bounded by $2\mathrm{e}\mu \ln(\mathrm{e}n/(n-L))$ if L does not increase before. If we sum up these expected waiting times for all values of L, we obtain (using Stirling's formula) a total expected waiting time of at most

$$2\mathrm{e}\mu \sum_{L=0}^{n-1} \ln\left(\frac{\mathrm{e}n}{n-L}\right) = 2\mathrm{e}\mu \ln\left(\frac{\mathrm{e}^n n^n}{n!}\right) \leq 2\mathrm{e}\mu \ln(\mathrm{e}^{2n}) = 4\mathrm{e}\mu n.$$

After the desired number of replicas has been obtained, the expected time for increasing L is at most $\frac{\mathrm{e}\mu n}{\min\{\mu,n/(n-L)\}\cdot(n-L)} = \frac{\mathrm{e}\mu n}{\min\{\mu(n-L),n)\}}$. By elementary calculations, the expected waiting time for all L-increases is at most $\mathrm{e}n \ln(\mathrm{e}n) + \mathrm{e}\mu n$, and the total expected runtime, therefore, at most $\mu+\mathrm{e}n \ln(\mathrm{e}n)+5\mathrm{e}\mu n$. □

For SPC, we can only prove a (seemingly) trivial upper bound.

Theorem 3. *Let $\mu = poly(n)$. Then the expected runtime of the (μ+1) EA on* SPC *is bounded by $O(\mu n^3)$.*

Sketch of proof. For each individual x from the initial population, we consider paths in its family tree directed from the root to a node v. If the individual

corresponding to v has been deleted, we call the path dead, and alive otherwise. There is always at least one alive path in some family tree.

We want to show that the following property P holds for every initial individual x. The expected time until at least one of x's paths reaches length k or until all of its paths are dead is bounded by $4e\mu k$ for all k. This will imply the theorem for the following reasons. By similar arguments as in the proof of Theorem 1, one can show that the (μ+1) EA reaches a situation where the entire population contains individuals of shape 1^i0^{n-i} with $i \neq n$, i. e., from the plateau of constant fitness, after $O(\mu n \log n)$ expected steps (or is done before). Afterwards, we can ignore steps of the (μ+1) EA creating individuals outside the plateau since these individuals are deleted immediately after creation. Since the (μ+1) EA chooses for deletion uniformly from the worst individuals, the event that a path dies is independent of the individual at the path's end provided it is from the plateau. Hence, any path of plateau points has the same properties as a path of plateau points drawn by a run of the (1+1) EA on SPC. By the results of Jansen and Wegener (2001a), such a path contains an optimal individual after an expected length of $O(n^3)$, i. e., after $O(\mu n^3)$ expected steps according to P.

To prove P, we assume w. l. o. g. that there is always at least one alive path for x. Consider the potential L, denoting the length of the currently longest alive path leading to an x' that will always have an alive descendant. There must be such an x' according to our assumptions. Moreover, L cannot shrink in the run, and there is the following sufficient condition for increasing L. An individual x' defining the current L-value is mutated, a child from the plateau is created, and x' is deleted before its child is deleted. The probability is $1/\mu$ for the first event, at least $(1-1/n)^n \geq 1/(2e)$ for the second event since producing a replica is sufficient, and $1/2$ for the third one since the considered individuals have equal fitness. Hence, the expected time to increase L is at most $4e\mu$, implying P. □

4 A General Lower Bound Technique

For lower bounds on the runtime, we consider the growth of the family tree for any initial individual of the (μ+1) EA. Upper bounds on the depth of family trees always follow from the selection mechanism of the (μ+1) EA, which selects the individual to be mutated uniformly from the current population. Therefore, it is possible to model the stochastic process growing a family tree as follows.

Definition 1 ($1/\mu$-tree). *Let $p := p_{t,u}$, $t, u \geq 0$, be a sequence of probability distributions s. t. the support of $p_{t,u}$ is $\{0, 1, \ldots, u\}$. A p-tree at time 0 consists only of the root. A p-tree T_t at time $t \geq 1$ is obtained from a p-tree T_{t-1} as follows. Let u be the number of nodes of T_{t-1}. Sample v by $p_{t-1,u}$. If $v > 0$, append a new leaf to the v-th inserted node of T_{t-1}; otherwise, let $T_t := T_{t-1}$.*

A p-tree is called a $1/\mu$-tree if $p_{t,u}(v) \leq 1/\mu$ for all $v > 0$.

A $1/\mu$-tree at time t can have less than $t+1$ nodes since p can put some probability on 0. If we model family trees by $1/\mu$-trees, we do not specify the distributions $p_{t,u}$ exactly since it is too difficult to predict whether and, if so,

which individuals corresponding to nodes are deleted. In the $(\mu+1)$ EA, deleted nodes have probability 0 of being chosen and alive nodes have probability $1/\mu$.

The following lemma contains an interesting result for the depth of $1/\mu$-trees.

Lemma 1. *Let $D(t)$ denote the depth of a $1/\mu$-tree at time t. For all $t \geq 0$ and $d \geq 0$,* $\text{Prob}(D(t) \geq d) \leq (t/\mu)^d/d!$. *Moreover,* $\text{Prob}(D(t) \geq 3t/\mu) = 2^{-\Omega(t/\mu)}$.

Lemma 1 states that, with overwhelming probability, a family tree of the $(\mu+1)$ EA becomes asymptotically no deeper than the total number of mutations performed in a single run of the corresponding parallel run. The tree can become wide, but a flat tree means that few mutations lie on any path from the root to a node in the tree. Hence, if the depth is small, this means that a leaf is an individual that is likely to be similar to the root. This makes the optimization of even simple functions very unlikely if the tree is not deep enough. The following result is tight for some simple functions such as OneMax (if μ is not too small).

Theorem 4. *Let $\mu = \text{poly}(n)$ and let f be a function with a unique global optimum. Then the expected runtime of the $(\mu+1)$ EA on f is $\Omega(\mu n + n \log n)$. Moreover, the success probability within some $c\mu n$ steps, $c > 0$, is $2^{-\Omega(n)}$.*

Sketch of proof. The lower bound of $\Omega(n \log n)$ follows for $\mu \leq \log n/2$ by a generalization of the coupon collector's theorem described by Droste, Jansen and Wegener (2002) for the considered class of functions and the (1+1) EA. For the lower bound $\Omega(\mu n)$, we set up a phase of length $s := \lfloor c\mu n \rfloor$ for some constant $c > 0$ and show that the $(\mu+1)$ EA requires at least s steps with probability $1 - 2^{-\Omega(n)}$ if c is small enough. The proof idea is as follows. In s steps, a family tree created by the $(\mu+1)$ EA with high probability has to reach a certain depth to optimize f; however, the probability of reaching this depth is very small.

Let x be an arbitrary initial individual x. We consider the infinite random process of building its family tree. Let $T_t(x)$ denote the tree at time t. According to Lemma 1, the probability of $T_s(x)'s$ depth reaching at least $3cn$ is $2^{-\Omega(n)}$. Now the aim is to prove that with probability $1 - 2^{-\Omega(n)}$, a depth of at least $3cn$ is necessary for optimization (if c is small enough).

During the process building the trees $T_t(x)$, we consider the event that a node v with optimal f-value is inserted. Consider the path p_v from x to v. We claim that with probability $1 - 2^{-\Omega(n)}$, its length is at least $n/4$. By Chernoff bounds (see Motwani and Raghavan (1995)), the root x has Hamming distance at least $n/3$ to the unique optimal string (represented by v) with probability $1 - 2^{-\Omega(n)}$. Moreover, consider a sequence of $n/4$ strings where each string is the result of a mutation of its predecessor by means of the $(\mu+1)$ EA's mutation operator. The expected Hamming distance of any two strings in this sequence is at most $n/4$, and, by Chernoff bounds, it is less than $n/3$ with probability $1 - 2^{-\Omega(n)}$. Since the nodes on each path in the trees $T_t(x)$ form such a random sequence of strings, the claim follows. Moreover, $T_s(x)$ contains at most $s = \text{poly}(n)$ paths, and there are at most polynomially many choices for x since $\mu = \text{poly}(n)$. Therefore, the probability that there is a node with optimal f-value at depth less than $n/4$ in a family tree at time s is still $2^{-\Omega(n)}$. If c is small

enough, $n/4$ is at least $3cn$. Since the sum of all failure probabilities is $2^{-\Omega(n)}$, the proof of the theorem is complete. □

Theorem 4 covers the wide range of unimodal functions. For some unimodal functions (e. g., linear functions), the (1+1) EA's expected runtime is $O(n \log n)$. For such functions, Theorem 4 states that the (μ+1) EA is (for large μ) at most by a factor of $O(\log n)$ more efficient than the corresponding parallel run.

For more difficult functions (meaning that the (1+1) EA's expected runtime is $\omega(n \log n)$), the proof concept of Theorem 4 can be carried over to show larger lower bounds also for the (μ+1) EA. However, we have to derive better lower bounds on the depth of family trees. Therefore, more structure of the function f and the encountered individuals comes into play. Although all nodes of a family tree are different individuals, many individuals may represent the same string $x \in \{0,1\}^n$. For an individual x^*, we call the $x \in \{0,1\}^n$ associated with x^* *the string of* x^* or say that x^* *is the string* x. We also call the string of an individual its *color*. This leads to the following definition.

Definition 2 (Monochromatic Subtree (MST)). *A connected subgraph of a family tree is called a* monochromatic subtree *if all its nodes are the same string.*

Obviously, all nodes in an MST have equal f-value. It is interesting that the stochastic process creating an MST sometimes equals the process for a so-called random recursive tree (RRT), a model of random trees well known from the literature (e. g., Smythe and Mahmoud (1995)). This will allow us to apply the known theory on RRTs. We obtain an RRT by the following stochastic process.

Definition 3 (Random Recursive Tree (RRT)). *An RRT at time* 0 *consists only of the root. An RRT* T_t *at time* $t \geq 1$ *is obtained from an RRT* T_{t-1} *by choosing uniformly at random one of its nodes and appending a new leaf to it.*

Note that the RRT at time $t \geq 0$ consists of exactly $t+1$ nodes. The processes generating MSTs and RRTs coincide only if the (μ+1) EA can choose uniformly from the set of nodes of the MST. Since deleted individuals are nevertheless kept in the family tree, this property can only be guaranteed if the individuals of the considered MST are still present in the population. To prove the following lemma, one exploits that considering MSTs, the event of appending a node whose color is different from that of the father is independent of the choice of the father.

Lemma 2. *Let* T^* *be a monochromatic subtree of a family tree and let* V *be the set of nodes of* T^**. If the (μ+1) EA does not delete any individual from* V *until the creation of the last node of* T^* *then* T^* *is an RRT.*

If the (μ+1) EA deletes individuals of an MST from the population, it chooses these, by the definition of an MST and the (μ+1) EA, uniformly from the alive nodes of the MST. Hence, the earliest inserted nodes have the highest chances of having been deleted by any fixed time t. Early inserted nodes are close to the root. This implies that an MST that is affected by deletion steps is typically deeper than an RRT of the same size. We can make this precise by considering generalized RRTs, namely so-called p-marked random trees (p-marked RTs).

Definition 4 (p-marked RT). *Let $p_{t,u}$, $t, u \geq 0$, be a sequence of probability distributions s. t. the support of $p_{t,u}$ is $\{0, \ldots, u\}$. A p-marked RT at time 0 consists only of the unmarked root. A p-marked RT T_t at time $t \geq 1$ is obtained from a p-marked RT T_{t-1} in two steps. First, an unmarked node is chosen uniformly at random and a new, unmarked leaf is appended. Let U denote the set of unmarked nodes after this step. Then u^* is sampled according to $p_{t-1,|U|-1}$, a subset $S^* \subseteq U$ of size u^* is chosen uniformly, and all nodes in S^* are marked.*

Again, a tree at time t has exactly $t + 1$ nodes, only the unmarked ones of which can become fathers of new nodes. It is crucial that for all $p_{t,u}$, the set of newly marked nodes is, by definition, uniform over the yet unmarked ones and that always at least one node remains unmarked.

Lemma 3. *A monochromatic subtree of a family tree is a p-marked RT.*

By technical analyses, one can show that the probability of a p-marked RT with t nodes reaching depth d is, for any p, at least as large as the respective probability of an RRT. Let for a p-marked RT and an RRT at time t the measures $D^*(t, i)$ resp. $D(t, i)$ denote the depth of the node that was inserted at time i.

Lemma 4. *For all $t, i, d \geq 0$ and $i \leq t$, $\text{Prob}(D^*(t, i) \geq d) \geq \text{Prob}(D(t, i) \geq d)$.*

Since lower bounds on the depth of ordinary RRTs are well known (Pittel (1994)), we have developed new tools for lower bounding the depth of MSTs and, therefore, of family trees. Upper bounds are still provided by Lemma 1.

5 More Special Lower Bounds

We apply the proof method developed in the last section to a well-studied function. Here, the method can also be considered as a generalization of the proof method of artificial fitness layers (e. g., Droste, Jansen and Wegener (2002)).

Theorem 5. *Let $\mu = poly(n)$. Then the expected runtime of the (μ+1) EA on* LeadingOnes *is $\Omega(\mu n \log n + n^2)$. Moreover, the success probability within some $c\mu n \log n$ steps, $c > 0$, is $2^{-\Omega(n)}$.*

Sketch of proof. The bound $\Omega(n^2)$ follows by applying the analysis of LeadingOnes and of the (1+1) EA by Droste, Jansen and Wegener (2002) to the potential L from the proof of Theorem 1. The basic idea for the bound $\Omega(\mu n \log n)$ is the same as in Theorem 4. We show that for some small enough constant $c > 0$, the (μ+1) EA requires at least $s := \lfloor c\mu n \log n \rfloor$ steps with probability $1 - 2^{-\Omega(n)}$. Now we consider the family tree $T_s(x)$ obtained after s steps for an arbitrary initial individual x. By Lemma 1, it suffices to show that a depth of at least $3cn \log n$ is necessary for optimization with probability $1 - 2^{-\Omega(n)}$.

For notational convenience, let $f :=$ LeadingOnes. During the process of building the trees $T_t(x)$, we consider the event that a node v with optimal f-value is inserted. Since initial individuals are uniform over $\{0, 1\}^n$, the root x has an

f-value of at most $n/2$ with probability $1-2^{-\Omega(n)}$. Consider the path p_v from x to v. By standard arguments from the analysis of the (1+1) EA on f (Droste, Jansen and Wegener (2002)), the bits after the leftmost zero are, in each string on p_v, uniformly distributed. W. l. o. g., the f-value is non-decreasing along p_v. Since the f-value has to increase by at least $n/2$ along p_v with probability $1-2^{-\Omega(n)}$, the mentioned arguments imply that at least $n/6$ different strings lie on p_v with probability $1-2^{-\Omega(n)}$. We call the nodes that are different strings than their fathers *subtree roots*. For a subtree root r, by $T^*(r)$ we denote the maximal MST rooted at r. Now we work under the assumption that p_v contains at least $n/6$ subtree roots.

Fix an arbitrary subtree root $r \neq v$ and the next subtree root r' on p_v. By Lemma 3, the MST $T^*(r)$ is a p-marked RT, and r' is some node inserted into (but not not attributed to) a p-marked RT. Considering the construction of $T^*(r)$, we prove that r' was likely to be created late during this process. The probability of mutating a string with value $f(r)$ to a better string is bounded above by $1/n$. Hence, with probability at least $1/2$, the first $n/2$ steps that choose a father in the already existing MST create nodes with at most the same value as the root. Since producing a replica of a string has probability $(1-1/n)^n \geq 1/(2\mathrm{e})$, the expected number of replicas within $n/2$ steps is at least $n/(4\mathrm{e})$. By Chernoff bounds, with probability at least $1/2-2^{-\Omega(n)}$, $T^*(r)$ receives at least $n/(8\mathrm{e})$ nodes before an individual with larger value than $f(r)$ is appended. Hence, with probability at least $1/2-2^{-\Omega(n)}$, the node r' has a distance to r that is bounded below by the depth of the at least $n/(8\mathrm{e})$-th node of a p-marked RT.

How deep is the k-th node such that $k \geq n/(8\mathrm{e})$ within a p-marked RT? We know it if the tree is an ordinary RRT. Then, by the theory on RRTs (Smythe and Mahmoud (1995)), the depth is at least $(\log n)/2$ with probability at least $1/2$ (for n large enough). By Lemma 4, the same statement holds also for a p-marked RT. Altogether, the distance of r and r' on p_v is at least $(\log n)/2$ with probability at least $1/4-o(1)$. Since the process creating $T^*(r')$ is independent of the process creating $T^*(r)$, we can apply Chernoff bounds. Since at least $n/6$ choices for r are available on p_v, at least $n/25$ subtree roots have their successive subtree roots at distance at least $(\log n)/2$ with probability $1-2^{-\Omega(n)}$. Altogether, the length of p_v is at least $n(\log n)/50$ with probability $1-2^{-\Omega(n)}$.

Since at time s, the number of all nodes in all trees is bounded by a polynomial, the probability that there is a node with f-value n at depth less than $n(\log n)/50$ in a family tree is $2^{-\Omega(n)}$. If c is small enough, the bound $n(\log n)/50$ is at least $3cn\log n$. Finally, the sum of all failure probabilities is $2^{-\Omega(n)}$. □

The method of lower bounding the depth of MSTs can also be used to lower bound the expected runtime of the (μ+1) EA on the function SPC. It is easy to see that, due to the plateau of constant fitness, there are with high probability even $\Omega(n^2)$ subtree roots on any path leading to an optimal node in a family tree. Hence, a straightforward application of the proof of Theorem 5 would lead to a lower bound of $\Omega(\mu n^2 \log n)$ on the expected runtime. However, one can improve on this by considering the number of alive nodes in MSTs (which a p-marked RTs and at least as deep as ordinary RRTs) more carefully. One can

show that p-marked RTs become the deeper the less alive nodes they contain. Considering SPC, one can analyze the random walk describing the number of alive individuals in MSTs. As with LeadingOnes, $\Theta(n)$ expected nodes are added to an MST before the first relevant node with different color is created. Since the probability of deleting a node from and of adding a node to an MST are almost equal, we can bound the number of alive nodes before this creation by $O(n^{1/2+\varepsilon})$ with high probability. This leads to a depth of $\Omega(n^{1/2-\varepsilon})$ with probability $\Omega(1)$. One can even refine this analysis to show an $\Omega(n^{1-\varepsilon})$ bound.

Theorem 6. *Let $\mu = poly(n)$. Then the expected runtime of the $(\mu+1)$ EA on* SPC *is $\Omega(\mu n^{3-\varepsilon})$ for any constant $\varepsilon > 0$. Moreover, the success probability within some $c\mu n^{3-\varepsilon}$ steps, $c > 0$, is $2^{-\Omega(n^{\varepsilon/4})}$.*

6 An Example Where $\mu > 1$ Is Essential

In the previous sections, we have shown for example functions that the $(\mu+1)$ EA can only be slightly more efficient than its corresponding parallel run. Moreover, it is never more efficient than a single run of the (1+1) EA on two of these functions, and it becomes less and less efficient for increasing values of μ.

However, it is believed that populations help to better explore search spaces. We can make this precise in some respect for an example function similar to that considered by Witt (2003) for a GA with fitness-proportional selection. Suppose that in a subspace $\{0,1\}^\ell$ of the search space, an optimal setting for LeadingOnes is sought, while in the subspace $\{0,1\}^{n-\ell}$, the optimum for OneMax is sought. If ℓ is not too small, the (1+1) EA normally finds the optimal setting for OneMax faster than for LeadingOnes. On the other hand, by the results from Sections 3–5, the expected runtime of the $(\mu+1)$ EA is $O(\mu\ell \log n + \ell n)$ for the LeadingOnes part and $\Omega(\mu(n-\ell))$ for the OneMax part. For $\ell = \sqrt{n}$ and $\mu = \Omega(n)$, e. g., this means that the $(\mu+1)$ EA is faster on the LeadingOnes part. This can be explained since now the subspace of the OneMax part is better explored but less exploited than the other subspace. If the function leads to an isolated local optimum if the OneMax part is optimized first, the (1+1) EA is expected to behave inefficiently. Moreover, if a global optimum is reached if the LeadingOnes part is optimized first, we expect the $(\mu+1)$ EA to be efficient.

The following function has been defined according to this idea. Let strings $x \in \{0,1\}^n$ be divided into a prefix $(x_1, \ldots, x_m)$ of length m and a suffix $(x_{m+1}, \ldots, x_n)$ of length ℓ. Let $\ell := \lceil n^{1/2} \rceil$, i. e., $m = n - o(n)$. For $x \in \{0,1\}^n$, we define $\mathrm{PO}(x) := x_1 + \cdots + x_m$ as the number of so-called **p**refix **o**nes. Let $\mathrm{LSO}(x) := \sum_{i=0}^{\ell-1} \prod_{j=0}^{i} x_{m+1+j}$ be the number of **l**eading **s**uffix **o**nes. Finally, let $b := 2m/3 + \lceil n^{1/2}/(700 \log^2 n) \rceil$. Then let

$$f(x) := \begin{cases} \mathrm{PO}(x) + n^2 \cdot \mathrm{LSO}(x) & \text{if } \mathrm{PO}(x) \leq 2m/3, \\ n^2\ell - n \cdot |\mathrm{PO}(x) - b| + \mathrm{LSO}(x) & \text{otherwise.} \end{cases}$$

We discuss the structure of f. The first case occurs if x has few POs. Then the f-value is strongly influenced by the number of LSOs. The optimum f-value of

$n^2\ell + 2m/3$ holds if $\mathrm{LSO}(x) = \ell$ and $\mathrm{PO}(x) = 2m/3$. However, if $\mathrm{PO}(x) \leq 2m/3$ and $\mathrm{LSO}(x) < \ell$, the f-value is at most $n^2(\ell - 1) + 2m/3$, which is less than $n^2\ell - nb$, a lower bound on the value in the second case ($\mathrm{PO}(x) > 2m/3$). If $\mathrm{PO}(x) = b$ and $\mathrm{LSO}(x) = \ell$, we have a locally optimal string with f-value $n^2\ell+\ell$. The Hamming distance to any better string is $b - 2m/3 = \Omega(n^{1/2}/\log^2 n)$. In fact, the (1+1) EA is likely to get stuck here, and even multistarts do not help.

Theorem 7. *With probability* $1 - 2^{-\Omega(n^{1/2}/\log n)}$*, the runtime of the (1+1) EA on* f *is* $2^{\Omega(n^{1/2}/\log n)}$.

Sketch of proof. We show that the (1+1) EA is likely to create b POs before ℓ LSOs. Then it has to overcome a Hamming distance at least $b - 2m/3$ in one step to reach the optimum. This takes $2^{\Omega(n^{1/2}/\log n)}$ steps with high probability.

We estimate the probability p^* of creating ℓ LSOs before reaching b POs as follows. With high probability, $O(n)$ steps suffice to create $b = m - \Omega(n)$ POs whereas increasing the LSO-value takes $\Omega(n)$ steps with probability $\Omega(1)$. Since $\Omega(n^{3/2})$ steps are necessary for ℓ LSOs with high probability, p^* is very small. □

Theorem 8. *Let* $n/\ln(en) \leq \mu = \mathit{poly}(n)$*. With probability* $1 - 2^{-\Omega(n^{1/2}/\log n)}$*, the runtime of the (*$\mu$*+1) EA on* f *is* $O(\mu n^{3/2}/\log n)$*. Its expectation is* $O(\mu n)$.

Sketch of proof. For the first claim, we use the idea of Theorem 1. Assume all individuals to have always at most $2m/3$ POs. Then we use the potential L, denoting the maximum number of LSOs in the population. By the definition of f, L cannot decrease, and no individual with L LSOs can be deleted if there are individuals with less LSOs. Hence, as a corollary of Theorem 1 for our choice of μ, the expected time until creating an individual with ℓ LSOs is at most $3e\mu\lceil n^{1/2}\rceil \ln(en)$. Moreover, is is easy to see that the time is $O(\mu n)$ with probability $1 - 2^{-\Omega(n^{1/2}/\log n)}$. Afterwards, there is always at least one individual with ℓ LSOs in the population. It is sufficient to reach the optimum by increasing the number of POs of such an individual to m. The expected time for this is bounded by $O(\mu n)$, and the time is $O(\mu n^{3/2}/\log n)$ with probability $1 - 2^{-\Omega(n^{1/2}/\log n)}$.

We estimate the probability that no individual ever has more than $2m/3$ POs within $s := \lfloor c\mu n\rfloor$ steps, $c > 0$ a constant, using the approach from Sect. 4. By Lemma 1, no family tree reaches a depth of at least $3cn$ with probability $1 - 2^{-\Omega(n)}$. No initial individual has at least $7m/12$ POs with probability $1 - 2^{-\Omega(n)}$. If c is chosen small enough, the probability of $\lceil 3cn\rceil$ mutations flipping a total number of at least $m/12$ bits is $2^{-\Omega(n)}$. Altogether, the probability of more than $2m/3$ POs within s steps is $2^{-\Omega(n)}$. Since the sum of all considered failure probabilities is $2^{-\Omega(n^{1/2}/\log n)}$, this proves the theorem's first statement.

For the statement on the expected runtime, we have to consider the case that an individual has more than $2m/3$ POs at some time. It is easy to see that then a locally optimal individual is created after $O(\mu n)$ expected steps. Since the Hamming distance to a locally optimal individual is bounded by $b - 2m/3 \leq n^{1/2}/(700\log^2 n) + 1$, the expected time until overcoming this distance is at most

$2^{(n^{1/2}+o(1))/(700 \log n)}$. The constants have been chosen such that the product of the waiting time and the probability of reaching the local optimum is $o(1)$. □

Conclusions

We have presented a first analysis of the (μ+1) EA for pseudo-Boolean functions by studying the expected runtime on three well-known example functions. For two of these, we have derived asymptotically tight bounds, showing that $\mu = 1$ leads asymptotically to the lowest runtime. In contrast to this, we have identified a function where the (μ+1) EA outperforms the (1+1) EA and its multistart variants drastically provided that $\mu \geq n/\ln(en)$.

To prove lower bounds, we have developed a new technique. This technique is not only limited to the (μ+1) EA. The upper bounds on the depth of family trees are independent of the mutation operator and even of the search space, and the lower bounds derived in the proofs of Theorem 4 and Theorem 5 hold for every selection operator choosing uniformly from individuals of the same fitness. For different selection-for-reproduction mechanisms, the concept of $1/\mu$-trees can be adapted. Nevertheless, the most interesting direction seems to be an extension to (μ+λ) strategies by a combination with the existing theory on the (1+λ) EA.

Acknowledgements. Thanks to Thomas Jansen and Ingo Wegener for discussions on the proof techniques and to the anonymous referees for helpful comments.

References

1. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science **276** (2002) 51–81
2. Droste, S., Jansen, T., Tinnefeld, K., Wegener, I.: A new framework for the valuation of algorithms for black-box optimization. In: Proc. of Foundations of Genetic Algorithms 7 (FOGA 2002), Morgan Kaufmann (2003) 253–270
3. Garnier, J., Kallel, L., Schoenauer, M.: Rigorous hitting times for binary mutations. Evolutionary Computation **7** (1999) 173–203
4. Giel, O.: Expected runtimes of a simple multi-objective evolutionary algorithm. In: Proc. of the 2003 Congress on Evol. Computation, IEEE Press (2003) 1918–1925
5. He, J., Yao, X.: From an individual to a population: An analysis of the first hitting time of population-based evolutionary algorithms. IEEE Transactions on Evolutionary Computation **6** (2002) 495–511
6. Jansen, T., De Jong, K.: An analysis of the role of offspring population size in EAs. In: Proc. of GECCO 2002. (2002) 238–246
7. Jansen, T., Wegener, I.: Evolutionary algorithms – how to cope with plateaus of constant fitness and when to reject strings of the same fitness. IEEE Transactions on Evolutionary Computation **5** (2001a) 589–599
8. Jansen, T., Wegener, I.: On the utility of populations. In: Proc. of GECCO 2001. (2001b) 1034–1041

9. Jansen, T., Wegener, I.: Real royal road functions – where crossover provably is essential. In: Proc. of GECCO 2001. (2001c) 375–382
10. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambr. Univ. Press (1995)
11. Pittel, B.: Note on the heights of random recursive trees and random m-ary search trees. Random Structures and Algorithms **5** (1994) 337–348
12. Rabani, Y., Rabinovich, Y., Sinclair, A.: A computational view of population genetics. Random Structures and Algorithms **12** (1998) 313–334
13. Smythe, R.T., Mahmoud, H.M.: A survey of recursive trees. Theory of Probability and Mathematical Statistics **51** (1995) 1–27
14. Storch, T., Wegener, I.: Real royal road functions for constant population size. In: Proc. of GECCO 2003. (2003) 1406–1417
15. Wegener, I., Witt, C.: On the optimization of monotone polynomials by the (1+1) EA and randomized local search. In: Proc. of GECCO 2003. (2003) 622–633
16. Witt, C.: Population size vs. runtime of a simple EA. In: Proc. of the 2003 Congress on Evol. Computation. Volume 3., IEEE Press (2003) 1996–2003